УДК 004.942

РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ТЕСТИРОВАНИЯ В СИСТЕМЕ УПРАВЛЕНИЯ КОНФИГУРАЦИЯМИ ANSIBLE

Бубненков Н.А., студент гр. ПИТ-01, I курс
Научный руководитель: Кубанских О.В., к.ф-м.н., доцент
Брянский государственный университет
имени академика И.Г. Петровского
г. Брянск

Ansible является довольно простым инструментом автоматизации для подготовки ресурсов в облаке, управления конфигурацией, развертывания приложений, оркестрации взаимодействия между сервисами и других задач. Данный инструмент использует язык YAML для описания конфигураций, что делает его легко читаемым и понятным для разработчиков и системных администраторов.

Ansible также имеет большое сообщество пользователей, занимающихся в IT-среде, которые создают и поддерживают множество модулей, плагинов и ролей для упрощения автоматизации и управления инфраструктурой

Предположим, что у нас есть система с установленным Jenkins и мы хотим привязать к ней IaaC в виде Ansible. После изначальной установки Jenkins, необходимо настроить связку с Ansible.

Разработка тестов

Перед конфигурированием Jenkins для запуска тестов, необходимо создать сами тестовые скрипты. Было решено проводить проверку синтаксиса сценариев. Для проверки синтаксиса одного сценария, а также вывода списка всех его задач, можно использовать встроенные команды Ansible.

Был написан скрипт на языке Shell, который вызывает команды проверки для каждого из файлов сценария в корневой директории проекта. На начальном этапе работы тест сохраняется в локальную сборочную директорию.

Далее необходимо провести юнит-тестирование функции модуля генерации HTML-страницы. В данный момент интерес представляет функция проверки формата файла, блок-схема которой представлена на рисунке 1.

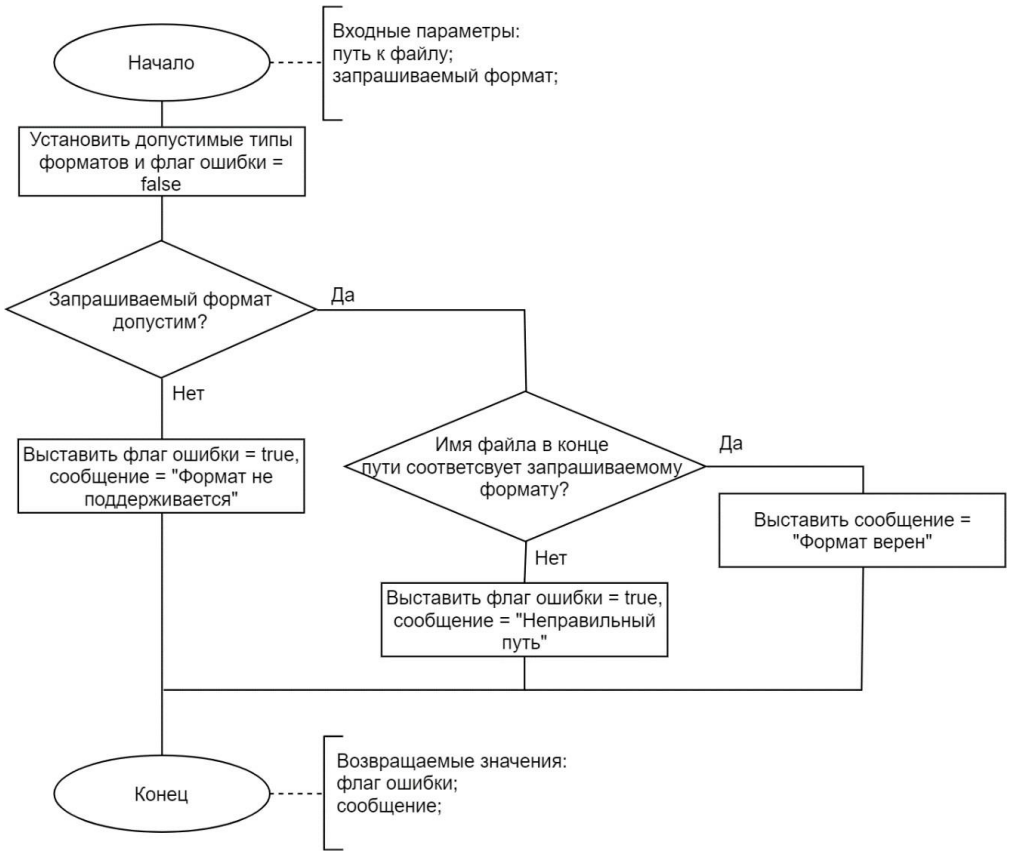

Рисунок 1 - Блок-схема алгоритма функции проверки формата файла

Для проверки работы функции можно подготовить три тестовых случая.

Тест будет подавать на вход функции четыре набора данных:
- верный путь к файлу, поддерживаемый формат;
- неверный путь к файлу, поддерживаемый формат;
- верный путь к файлу, неподдерживаемый формат;
- неверный путь к файлу, неподдерживаемый формат.

При этом тест будет ожидать следующие результаты работы функции:
- флаг ошибки false, сообщение «Формат верен»;
- флаг ошибки true, сообщение «Неправильный путь»;
- флаг ошибки true, сообщение «Формат не поддерживается»;
- флаг ошибки true, сообщение «Формат не поддерживается».

Если какой-либо ответ не совпадет с ожидаемым, тест будет считаться проваленным. Тест находится в одной директории с тестируемым Ansible-модулем.

Создание конфигурации для средства непрерывной интеграции

Следующим этапом конфигурирования средства непрерывной интеграции является создание всех необходимых образов и конфигурационных файлов для автоматизации запуска ранее написанных проверок синтаксиса

или юнит-тестов. Было решено запускать программный конвейер Jenkins в отдельном контейнере. Ранее при установке Jenkins была произведена настройка, позволяющая запускать отдельный контейнер из основного контейнера Jenkins. Изначально в контейнере Jenkins установлена ОС Debian. Для того, чтобы можно было проводить тесты на различных системах, было решено использовать дополнительные

контейнеры, в которых установлены другие системы.

При создании конфигурации первым делом создается контейнер с операционной системой CentOS7. Внутри образа помещается исходный код и тестирующий скрипт, созданный ранее. Блок-схема алгоритма представлена на рисунке 2.

Рисунок 2 - Блок-схема алгоритма подготовки конфигурации Jenkins

Первым делом создается сборочная директория, куда временно перемещается исходный код тестируемого проекта. Происходит генерация файлов Docker. В случае онлайн-режима происходит скачивание нового образа CentOS7, а в случае оффлайн режима используется существующий архив, из которого образ загружается в локальный репозиторий Docker. Далее, независимо от режима, происходит сборка образа тестирующей системы на основе CentOS7.

Передача и применение конфигурации на тестирующую машину

Передача собранного образа системы происходит аналогично передаче зависимостей и образов для Jenkins и Nginx.

В Jenkins для объявления последовательности действий наиболее часто используется понятие программного контейнера.

Программный конвейер (от английского «pipeline») – представление процесса разработки программного обеспечения в виде последовательности действий.

Плагины Jenkins, которые ранее были установлены, предоставляют возможность создавать такие программные конвейеры. Самый простой способ создания конвейера – его объявление через веб-интерфейс Jenkins. Настройка конвейера через веб-интерфейс показана на рисунке 3.

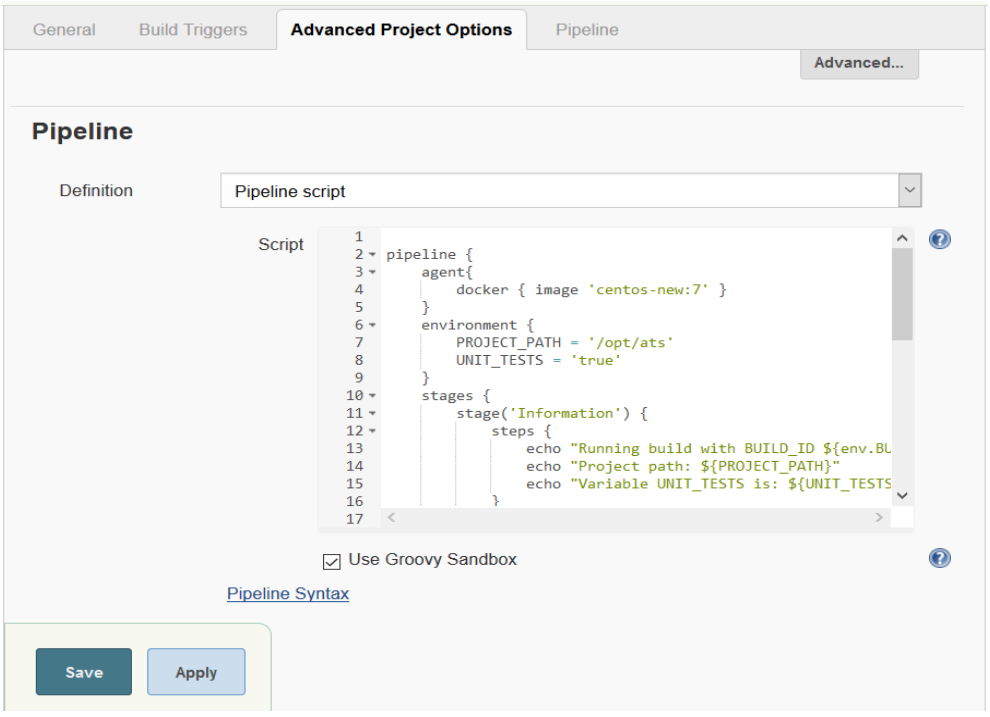

Рисунок 3 - Настройка конвейера через веб-интерфейс Jenkins

Однако, для того чтобы создавать конвейеры программным способом, необходимо создать шаблоны для конфигурационных файлов. Все конвейеры и задачи Jenkins хранятся в виде конфигурационных файлов.

Одним из способов автоматизации создания этих элементов являет-

ся копирование заранее заготовленных конфигурационных файлов в необходимую директорию. Для этого ранее, на этапе запуска Jenkins, были созданы общие директории между контейнером и виртуальной машиной.

Помещая конфигурационный файл в эту директорию, становится возможным создавать пользовательские конвейеры программным путем. Первая часть блок-схемы алгоритма конфигурирования Jenkins представлена на рисунке 4.

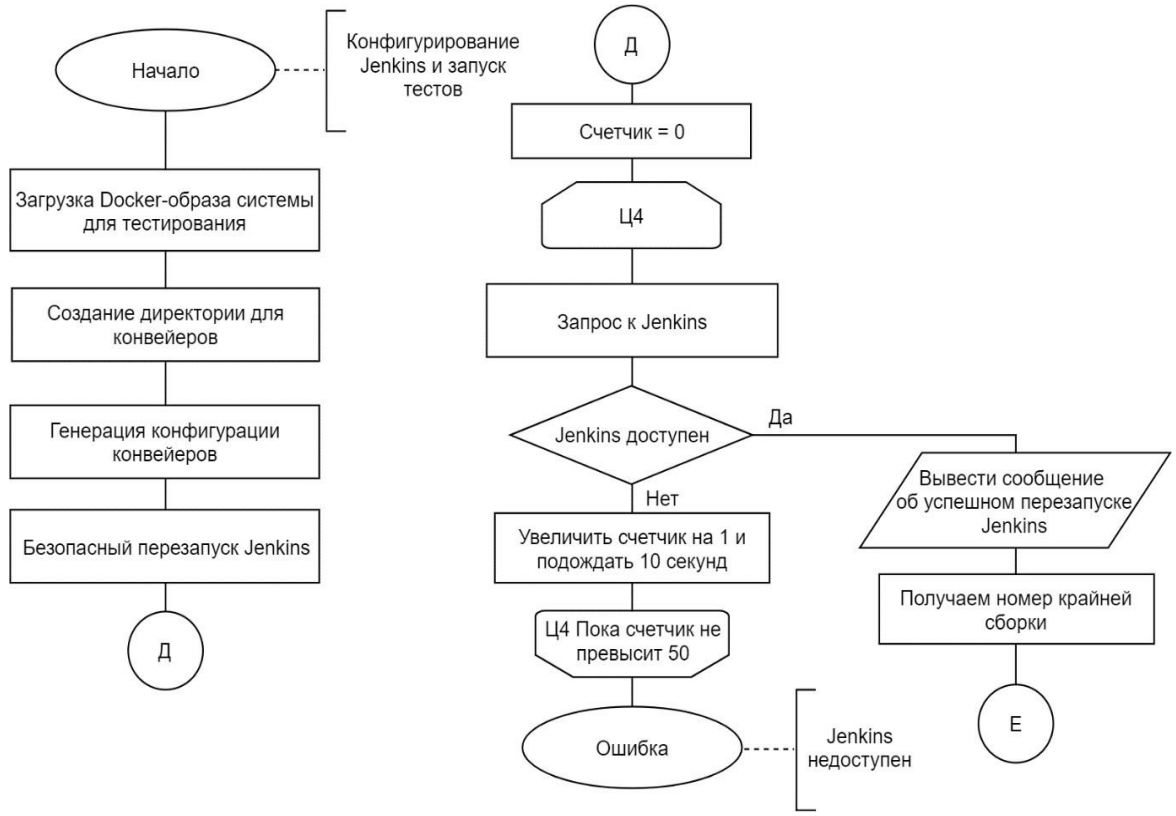

Рисунок 4 - Блок-схема алгоритма конфигурирования Jenkins и запуска тестов, часть 1

Первым делом, происходит загрузка образа тестируемой системы. В дальнейшем этот образ будет доступен из контейнера Jenkins. Далее происходит создание директории для конвейера. В эту директорию помещается конфигурацияконвейера.

Далее происходит перезапуск Jenkins и ожидается ответ от него. После успешного перезапуска Jenkins, новый конвейер становится доступным для запуска. Так как для запуска нового конвейера требуется время, то для того, чтобы убедиться в успешности его запуска, нужно сохранить номер предыдущей сборки конвейера (в случае первого запуска там будет пустое значение).

Вторая часть блок-схемы алгоритма конфигурирования Jenkins представлена на рисунке 5.

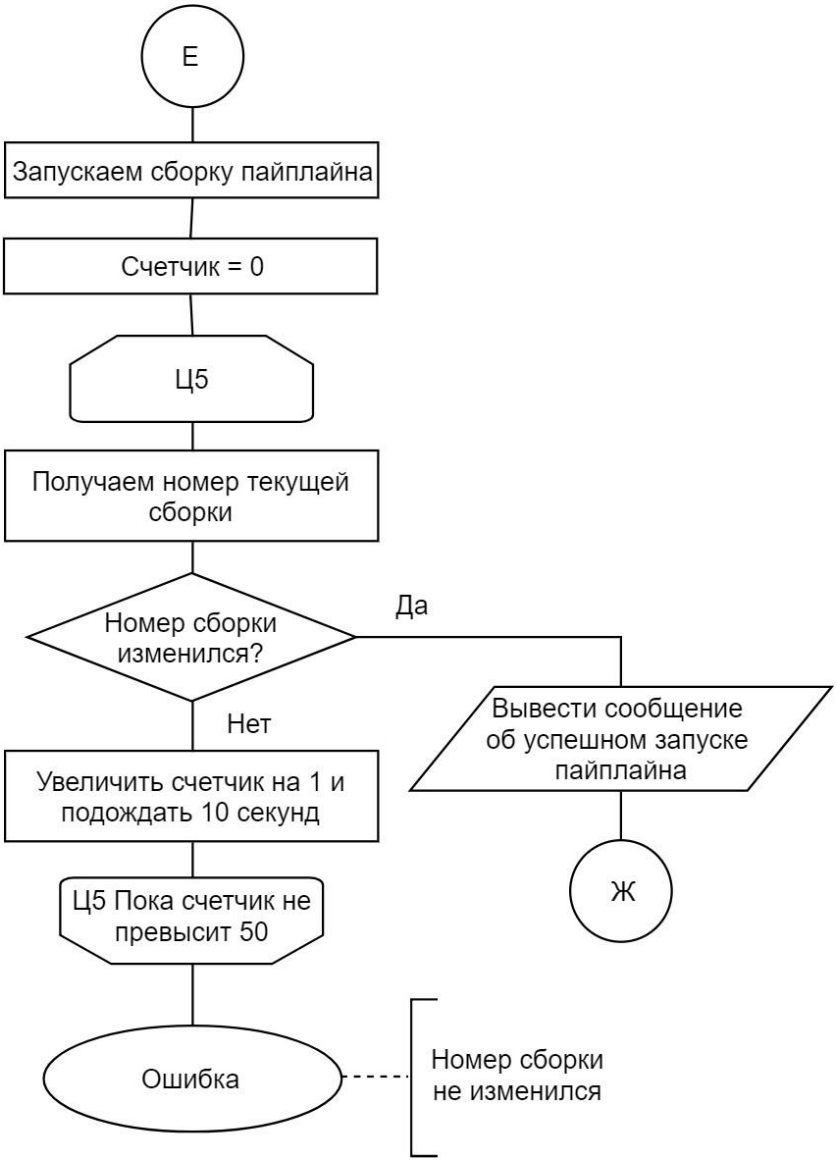

Рисунок 5 - Блок-схема алгоритма конфигурирования Jenkins и запуска тестов, часть 2

После этого происходит запуск конвейера с помощью запроса к программному интерфейсу. Далее в цикле ожидается изменение номера сборки. Когда номер сборки изменился, следующим этапом является ожидание результатов тестов. Третья часть блок-схемы алгоритма конфигурирования Jenkins представлена на рисунке 6.

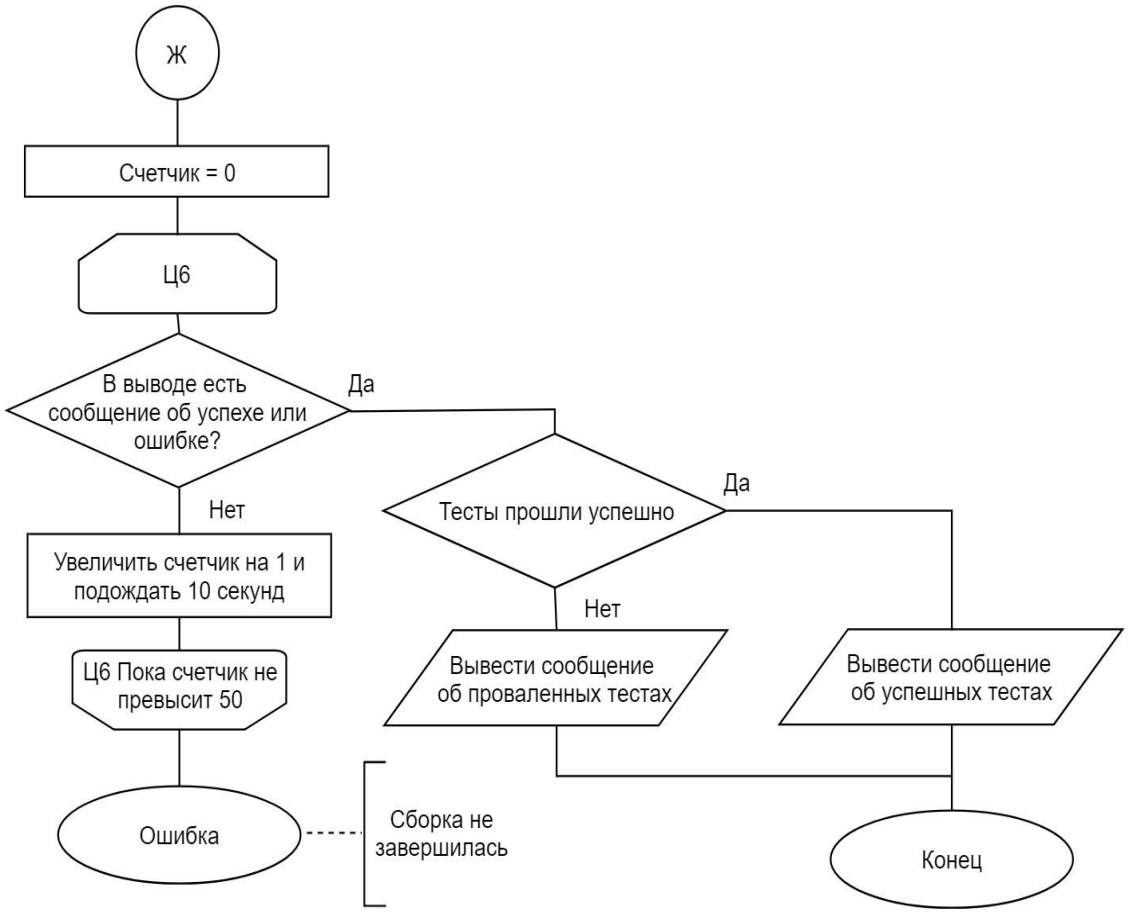

Рисунок 6 - Блок-схема алгоритма конфигурирования Jenkins и запуска тестов, часть 3

Текущие сообщения о процессе работы отображаются в специальную консоль, к которой есть доступ по программному интерфейсу Jenkins. Сценарий ожидает того, что в выводе тестов появятся сообщения об успешности или о провале тестов. Как только появляется сообщение, оно отображается на экране для пользователя. В случае провала теста, на экран выводится всё содержимое консоли для определения причин ошибки разработчиком.

На этом модульные тесты считаются оконченными.

Список литературы:

1. Basic Concepts // Ansible Documentation [Электронный ресурс]. URL: https://docs.ansible.com/ansible/latest/network/getting_started/basic_concepts.html (дата обращения: 26.03.2023).
2. CentOS7 Project [Электронный ресурс]. URL: https://www.centos.org/ (дата обращения 25.03.2023).
3. The Shell scripting tutorial [Электронный ресурс]. URL: https://www.shellscript.sh/ (дата обращения 23.03.2023).
4. Басыня Е.А., Лукина М.С. Автоматизированная установка и конфигурирование серверных решений / Е.А. Басыня, М.С. Лукина // Современные материалы, техника и технологии. – 2016. – №2. [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/avtomatizirovannaya-ustanovka-i-konfigurirovanie-servernyh-resheniy (дата обращения: 25.03.2023).
5. Иванова, Н.А. Создание корпоративной почтовой системы с использованием системы управления конфигурациями / Н.А. Иванова, О.В. Кубанских // Прикаспийский журнал: управление и высокие технологии. – 2022. – № 1 (57). – С. 17-26. – DOI 10.54398/2074-1707_2022_1_17.